DOI: 10.15587/2312-8372.2018.133694

ОПТИМІЗАЦІЯ АЦИКЛІЧНИХ СУМАТОРІВ БІНАРНИХ КОДІВ

Соломко М. Т.

Об'єктом дослідження є префіксна модель обчислення сигналів суми і перенесення у схемі паралельного суматора з паралельним способом перенесення. Одним з найбільш проблемних місць префіксної моделі є процес вироблення сигналів суми і перенесення, у якому початок обчислення префікса передбачено з першого розряду схеми. Це приводить, у підсумку, до надлишкового нагромадження і ускладнення апаратної частини пристрою.

У ході дослідження використовувалась математична модель обчислення сигналів суми і перенесення у схемі паралельного суматора, що ґрунтується на властивостях направленого ациклічного графа з двома типовими операціями.

Отримано зменшення складності логічної структури суматора бінарних кодів, зменшення глибини схеми та зменшення загальної протяжності з'єднувальних проводів. Це пов'язано з тим, що запропонований метод обчислення сигналів суми і перенесення має ряд особливостей синтезу схеми пристрою, зокрема застосування математичної моделі, що ґрунтується на властивостях ациклічного графа, розраховано на:

− процес послідовного (для молодших розрядів схеми пристрою) і паралельного обчислення сигналів суми і перенесення, що, у підсумку, дає зменшення складності апаратної частини пристрою та не збільшує глибину схеми;

− співставлення числа обчислювальних кроків орієнтованого ациклічного графа з числом перенесень одиниці до старшого розряду у схемі суматора, що дозволяє встановлювати оптимальне число обчислювальних кроків для структури пристрою.

Завдяки цьому забезпечується можливість отримання оптимальних значень показників складності структури та глибини схеми суматора. Зв'язок між числом обчислювальних кроків орієнтованого ациклічного графа і числом перенесень у схемі паралельного суматора з паралельним способом перенесення вказує на доцільність співставлення структури суматора з відповідним орієнтованим ациклічним графом.

У порівнянні з аналогічними відомими структурами 8-bit префіксних суматорів це забезпечує збільшення показника якості 8-bit ациклічних суматорів, наприклад, за енергоспоживанням, площею чіпа, у залежності від обраної структури, на 10–40 %.

Ключові слова: *ациклічна модель, префіксна модель, направлений ациклічний граф, Ling Adder, Kogge-Stone Adder, Brent-Kung Adder.*

1. Вступ

Суматор бінарних кодів присутній у більшості цифрових електронних схем, включаючи цифрові сигнальні процесори (DSP) і є одним із засобів

мікропроцесорної обробки даних. Продуктивність операції додавання у позиційній системі числення залежить від способу перенесення одиниці до старшого розряду. Варіантом такого перенесення є, зокрема, технологія префіксного підсумовування чисел [1–4].

У даній роботі представлено застосування ациклічної моделі обчислення сигналів суми і перенесення [5] для синтезу паралельних 8-bit суматорів бінарних кодів. Це дає новий апарат синтезу паралельних багаторозрядних суматорів з паралельним способом перенесення для застосування їх у цифрових технологіях.

Методи арифметичних операцій реалізуються вентильними схемами з функціональних елементів у базисах, що складаються з функцій алгебри логіки. Від структури суматора залежить швидкодія цифрового пристрою його надійність та енергозбереження. У зв’язку з цим мінімізація складності та глибини логічних схем є однією з центральних і практично важливих проблем у цій теорії, яка постає під час проектування цифрових пристроїв.

Процесорна еволюція є результатом невпинної оптимізації, тому актуальними залишаються дослідження направлені, зокрема, на вдосконалення таких чинників – технології виготовлення, структурної реалізації, величини енергоспоживання, вартості цифрових пристроїв.

2. Об'єкт дослідження та його технологічний аудит

Об’єктом вирішення задачі синтезу схеми суматора бінарних кодів є ациклічна модель одноетапного обчислення сигналів суми і перенесення, що ґрунтується на властивостях направленого ациклічного графа з двома типовими операціями (рис. 1, 2).

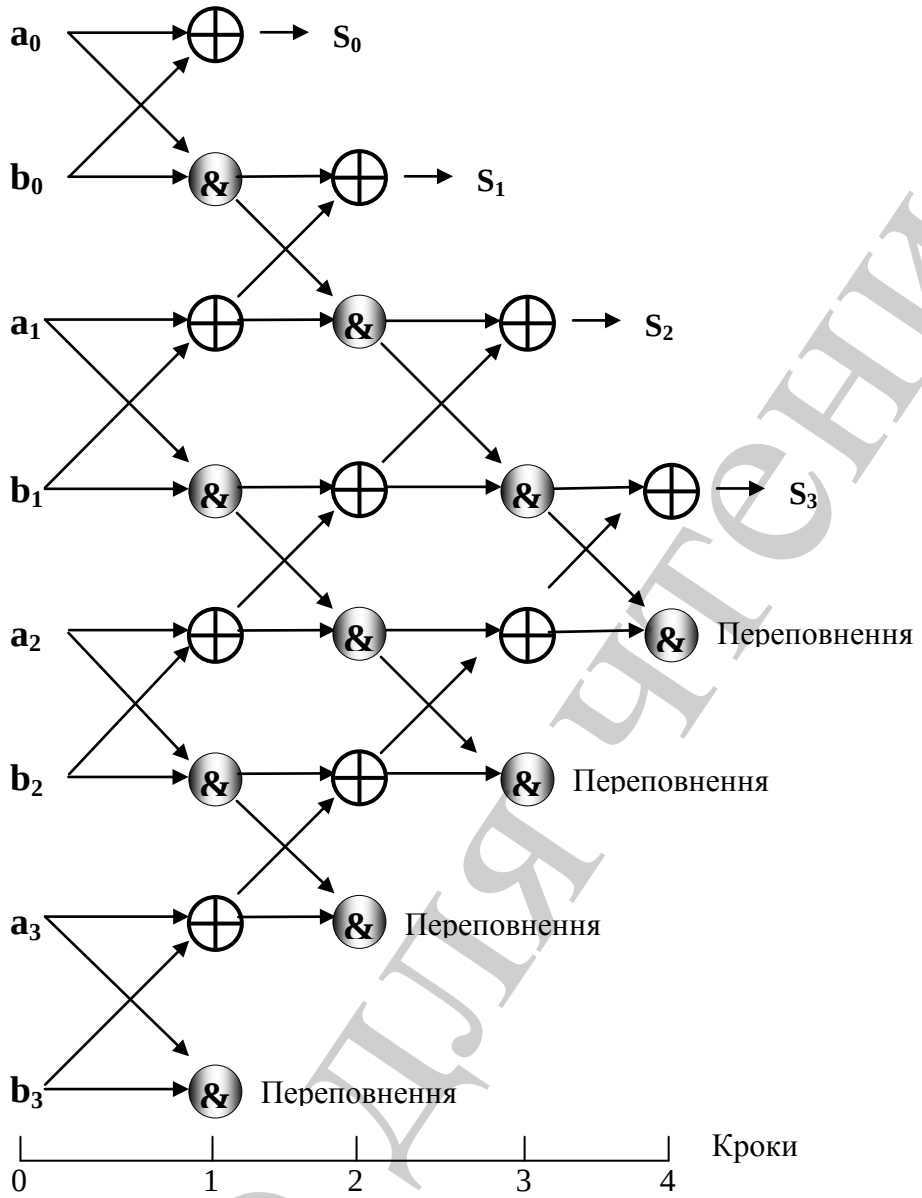

Рис. 1. Орієнтований ациклічний граф – модель обчислювальної схеми паралельного 4-розрядного ациклічного суматора з паралельним способом перенесення

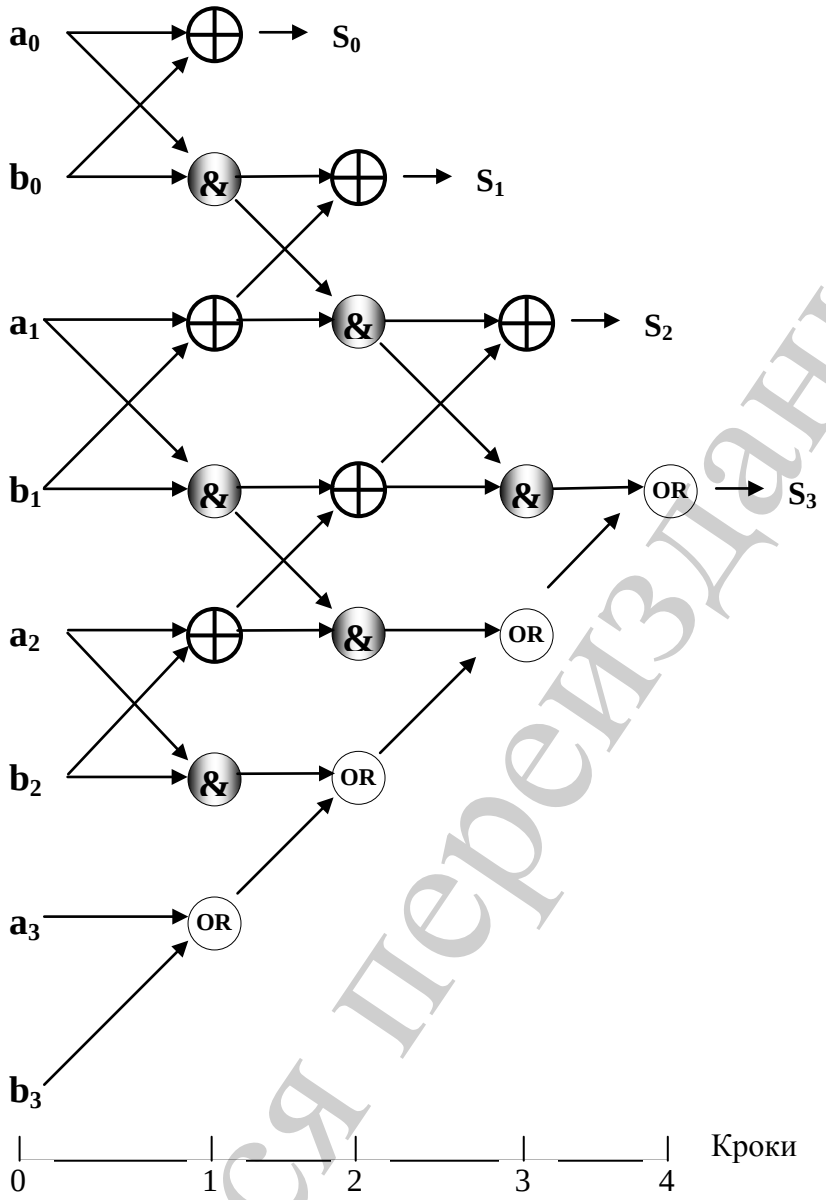

Рис. 2. Орієнтований ациклічний граф – модель обчислювальної схеми паралельного 4-розрядного ациклічного суматора з логічними елементами OR в останньому розряді

Ациклічна модель розрахована на логічну структуру суматора з послідовно-паралельним способом обчислення префікса, що, у підсумку, приводить до зменшення складності апаратної частини пристрою. Математичний апарат направленого ациклічного графа дозволяє однозначно отримувати значення сигналів суми і перенесення за один етап обчислення, тому останній спроможний з ефектом замінити трьох етапну префіксну модель обчислення сигналів суми і перенесення (рис. 3). Це розширює апарат синтезу арифметичних пристроїв для застосування їх у цифрових технологіях.

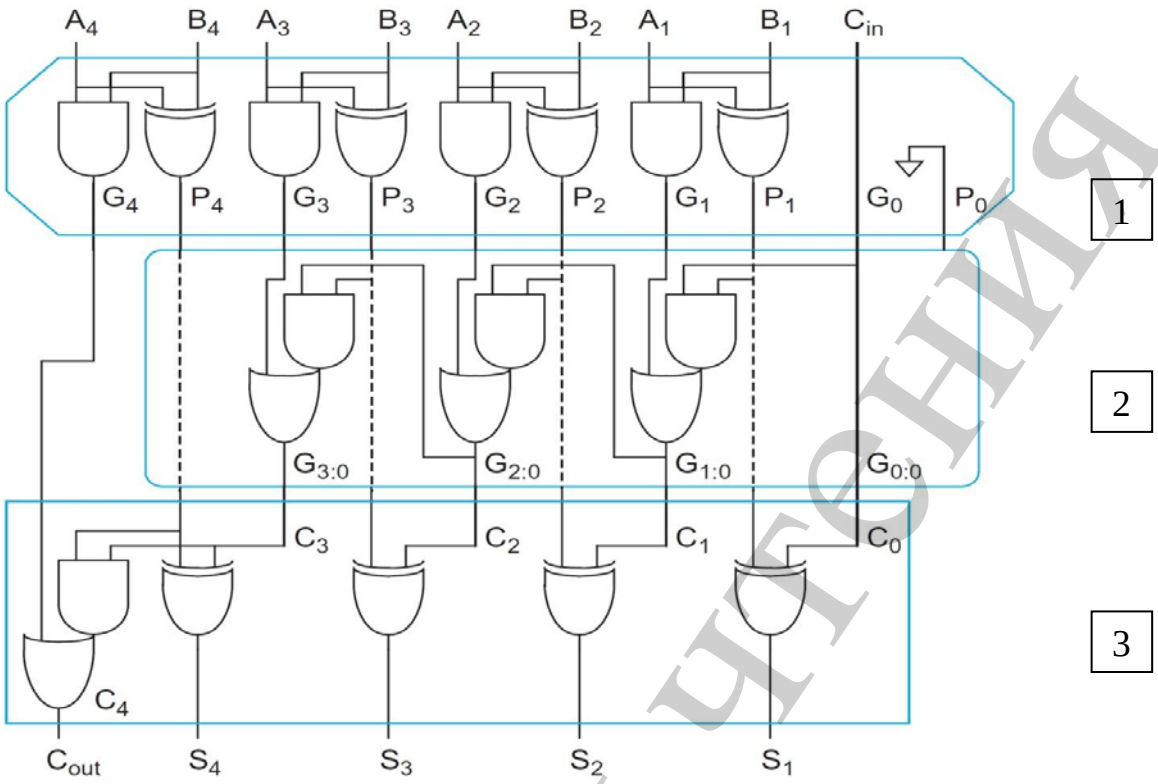

Рис. 3. Модель паралельного префіксного суматора:
1 – організаційна логіка; 2 – групова логіка; 3 – логіка суми бінарного коду

Ациклічна модель суматора представляється двома типовими операціями – AND і XOR, допускає способи застосування функції умови перенесення одиниці до старшого розряду (1), що у підсумку приводить до оптимальної складності схеми арифметичного пристрою.

$$p_i = a_i \vee b_i \quad \text{або} \quad p_i = a_i + b_i. \tag{1}$$

Ациклічна модель арифметичного пристрою спроможна підтримувати агреговані структури обчислення сигналів суми і перенесення, шляхом об'єднання з відповідним апаратом інших методів обчислення, зокрема з логікою перенесення Лінга.

Відносний недолік ациклічної моделі обчислення сигналів суми і перенесення на даний час пов'язаний з малим об'ємом теоретичних розробок за цим напрямком, тому перспектива методу ґрунтується на практичних шансах синтезу оптимальної структури арифметичного пристрою.

3. Мета та задачі дослідження

Метою роботи є синтез 8-bit оптимальних паралельних ациклічних суматорів бінарних кодів.

Для досягнення поставленої мети необхідно вирішити такі задачі:

1. Встановити адекватність математичної моделі на основі орієнтованого ациклічного графа з двома типовими операціями.

2. Оцінити динаміку збільшення глибини схеми паралельного ациклічного суматора зі збільшенням розрядності схеми.

3. Провести порівнянний аналіз складності структури та швидкодії суматорів, отриманих за допомогою ациклічної та префіксної моделей обчислення сигналів суми і перенесення.

4. Дослідження існуючих рішень проблеми

Каскадну схему, як механізм обчислення у складі префіксної моделі суматора, що використовує логічну структуру трьох етапного обчислення сигналів суми і перенесення (рис. 3), представлено у [6]. Зазначимо, що ациклічна модель обчислення сигналів суми і перенесення (рис. 1, 2) розрахована на логічну структуру суматора з послідовно-паралельним способом обчислення префікса та використовує структуру одноетапного обчислення. Таким чином, префіксна і ациклічна моделі є різними об'єктами – мають різні початки (принципи) обчислення, а від так володіють різними можливостями стосовно швидкодії обчислення, площі чіпа та енергозбереження.

З метою підвищення продуктивності обчислення у [7] досліджуються префіксні структури Kogge-Stone та Ladner-Fischer. Представлено порівняння з іншими схемами обчислень, зокрема з RCA, Carry Skip Adder (CSA). У проекті використовується інструмент Xilinx-ISE.

Модифікований Parallel Prefix Han-Carlson Adder представлено у [8], який використовує різні етапи синтезу структури префікса Brent-Kung і Kogge-Stone, що дає можливість зменшити складність дизайну суматора.

Способи зменшення затримки обчислення сигналів суми і перенесення у схемі суматора за допомогою структури паралельного префікса розглянуто у роботі [9], оскільки така структура попередньо обчислює перенесення. Розглянуто структури префікса Kogge-Stone та Brent-Kung. Процес моделювання та синтезу виконаний за допомогою моделі sim6.4b, Xilinx ISE9.2i.

Гібридну префіксну архітектуру для синтезу 8-, 16- та 32-bit паралельних суматорів представлено у роботі [10]. Проведені порівняння затримки, енергозбереження, числа обчислювальних вузлів з класичними префіксними структурами. Результати порівняння демонструють меншу затримку та енергоспоживання у запропонованих обчислювальних структурах. Для моделювання суматора за технологією 180 нм і 130 нм використано інструмент Tanner EDA.

Гібридну префіксну архітектуру Han Carlson Adder для зменшення енергоспоживання і затримки у паралельному префіксному суматорі (PPA) розглянуто в [11]. Проведено порівняння з іншими префіксними структурами.

У роботі [12] представлено розробку та порівняння високошвидкісних додаткових елементів префіксу, таких як Kogge-Stone, Brent-Kung, Sklansky та Ling. Виявлено, що структура Kogge-Stone-Ling є більш ефективною, порівняно

з іншими префіксними структурами. Проектування використовує логіку КМОП. Дизайн і моделювання виконано за допомогою 65-нм технології.

У роботі [13] зазначено, що кожен тип паралельного префіксного суматора має свої переваги і недоліки та обирається відповідно до вимог заявленого дизайну. У цій роботі досліджуються, головним чином, два типи структур, що містять комбіновані дерева і суматор Kogge-Stone та порівнюють їх. Проекти реалізовані на Xilinx Virtex 5 FPGA. З'ясовано, що комбіновані дерева займають меншу площу у порівнянні зі структурою Kogge-Stone.

Спосіб мінімізації енергоспоживання шляхом досягнення оптимальної структури паралельного Kogge-Stone та Ladner-Fischer суматора на 32-, 64-, 128- і 256-bit для 45-нм КМОП-технології досліджено у роботі [14]. Результати дослідження демонструють зменшення енергоспоживання на 22–50 % для оптимальної структури суматора з незмінною продуктивністю обчислення.

У роботі [15] зазначено, що префіксні структури є ефективними для реалізації ASIC, але цих переваг недостатньо для розробки FPGA. Представлені різні типи паралельних префіксів для порівняння та вибору. Для розробки додатків застосовано Verilog HDL, програмний засіб Xilinx ISE13.2 та компілятор Cadence RTL. Серед усіх додатків Kogge-Stone суматор забезпечує кращу продуктивність у реалізації ASIC, але це не підходить для розробки FPGA. Для того, щоб зробити його придатним для FPGA реалізації Kogge-Stone суматор модифікується за допомогою швидкої логіки, що забезпечує оптимальну продуктивність.

Патент [16] представляє пірамідальну структуру комбінаційного суматора з вертикальними і горизонтальними інформаційними зв'язками між однорозрядними двійковими напівсуматорами. Технічним результатом патенту є розширення функціональних можливостей пристрою, зменшення апаратної складності за рахунок введення швидкодіючих однорозрядних напівсуматорів, які містять три логічні елементи, та підвищення швидкодії пристрою.

На відміну від розглянутих публікацій у даній роботі об'єктом для синтезу структури суматорів бінарних кодів є ациклічна модель, опис якої наведено у розділі 2.

5. Методи дослідження

5.1. Префіксна модель суматора бінарних кодів

Префіксною сумою або просто префіксом послідовності чисел x_0, x_1, x_2, …, x_n є інша послідовність чисел y_0, y_1, y_2, …, y_n, яка обчислюється з вихідної за таким принципом:

$$y_0=x_0,$$
$$y_1=x_0+x_1,$$
$$y_2=x_0+x_1+x_2,$$
$$\ldots$$
$$y_n=x_0+\ldots+x_{n-1}+x_n.$$

У каскадному суматорі біт перенесення c_i обчислюється у момент часу i. Значення a_i і b_i відомі з самого початку. У деяких випадках вони визначають біт перенесення c_i:

якщо $a_i=b_i=0$, то $c_i=0$ (перенесення «поглинається» (kill)),
якщо $a_i=b_i=1$, то $c_i=1$ (перенос «породжується» (generate)).

Однак, якщо один з бітів a_i або b_i дорівнює 1, а інший 0, то c_{i-1} має суттєвий зміст для перенесення, тобто:

якщо $a_i \neq b_i$, то $c_i=c_{i-1}$ (перенесення розповсюджується (propagate)).

Кожному розряду, отже, відповідає один з трьох типів перенесення (carry statuses): k (kill), g (generate) або p (propagate). Цей тип відомий наперед, що дозволяє зменшити час проведення операції додавання.

Оскільки тип перенесення для сусідніх розрядів (($i-1$)-го та i-го) відомий, можна визначити тип перенесення для їх об'єднання, вважаючи c_{i-1} вхідним бітом, а c_{i+1} – вихідним. Таким чином, отримуючи інформацію про те, як змінюється біт перенесення на кожному кроці, можна розрахувати, що відбудеться за два кроки, тобто як залежить c_{i+1} від c_{i-1}. Якщо i-й розряд має тип перенесення p, то тип перенесення для об'єднання співпадає з типом ($i-1$)-го розряду (табл. 1).

Таблиця 1

Таблиця операції Θ

Θ		FA_i		
		k	p	g
FA_{i-1}	k	k	k	g
	p	k	p	g
	g	k	k	g

Табл. 1 можна розглядати як визначення операції (композиції типів перенесення) на множині $\{k, p, g\}$; вона позначається символом Θ і є асоціативною. Операція Θ визначає тип перенесення для деякої ділянки суматора, якщо відомі типи перенесень окремих його розрядів.

Позначимо через x_i тип перенесення в i-му розряді так:

$$x_i = \begin{cases} k, \text{ якщо } a_i=b_i=0; \\ g, \text{ якщо } a_i=b_i=1; \\ p, \text{ якщо } a_i \neq b_i. \end{cases}$$

Тоді залежність, наприклад, біта c_7 від c_4 визначається композицією:

$x_5 \Theta x_6 \Theta x_7$.

Оскільки перенесення одиниці до нульового розряду від молодших розрядів не здійснюється, умовно приймається $x_0=k$. Тоді перенесення на виході i-го розряду визначається композицією $x_0 \Theta x_1 \ldots \Theta x_i : c_i=0$, якщо композиція дорівнює k, і $c_i=1$, якщо композиція дорівнює g. Значення p для композиції неможливе, оскільки для цього всі члени повинні бути рівними p, а це не так для x_0.

Більш формально це записується так. Приймемо $y_0=k$ і визначимо $y_1, y_2, \ldots, y_n$, у вигляді:

$$y_i = x_i \Theta y_{i-1} = x_0 \Theta x_1 \Theta \ldots \Theta x_i.$$

Тоді $y_1, y_2, \ldots, y_n$ є префіксами (prefixes) виразу $x_0 \Theta x_1 \ldots \Theta x_n$.

Таким чином, обчислення суми бітів перенесення одиниці c_i у каскадному суматорі можна звести до обчислення префіксів.

Метод паралельного префіксу виник як найшвидший процес підсумовування бітів перенесення одиниці c_i в операціях додавання бінарних кодів для високопродуктивних систем обробки даних, оригінальні ідеї якого можна знайти у ранніх роботах [1–4]. Подальші публікації [17–20] підтвердили зазначену оцінку таких технологій.

При визначенні перенесення у паралельному багаторозрядному суматорі принцип отримання префіксної суми на послідовності чисел $x_0, x_1, x_2, \ldots, x_n$ поширюється на отримання префіксної суми на послідовності пар функцій перенесення одиниці $(g_0, p_0), (g_1, p_1), \ldots, (g_{k-1}, p_{k-1})$ (табл. 2):

$$g_i = a_i b_i \text{ – функція генерації перенесення;}$$
$$p_i = a_i \oplus b_i \text{ – функція розповсюдження перенесення.} \qquad (2)$$

Таблиця 2

Обчислення префіксної суми на послідовності пар функцій перенесення одиниці $(g_0, p_0), (g_1, p_1), \ldots, (g_{k-1}, p_{k-1})$

Дано	(g_0, p_0)	(g_1, p_1)	…	(g_{k-2}, p_{k-2})	(g_{k-1}, p_{k-1})
Знайти	$(g_{[0,0]}, p_{[0,0]})$	$(g_{[0,1]}, p_{[0,1]})$	…	$(g_{[0,k-2]}, p_{[0,k-2]})$	$(g_{[0,k-1]}, p_{[0,k-1]})$
Перенесення одиниці на виході i-го розряду	c_1	c_2	…	c_{k-1}	c_k

Перенесення одиниці на виході i-го розряду визначається композицією:

$$(g_0, p_0) \Theta (g_1, p_1) \Theta \ldots \Theta (g_{k-2}, p_{k-2}) \Theta (g_{k-1}, p_{k-1}).$$

Оператор, що визначає перенесення Θ, є асоціативним, але не комутативним:

$$[(g_1, p_1)\Theta(g_2, p_2)]\Theta(g_3, p_3)=(g_1, p_1)\Theta[(g_2, p_2)\Theta(g_3, p_3)].$$

Функцію розповсюдження перенесення (1) більш точно можна назвати функцією умови перенесення одиниці до старшого розряду. Часто функція розповсюдження перенесення (2) (умова перенесення) визначається як функція (1).

Якщо $p_i = 1$, то розповсюдження одиниці до наступних розрядів буде можливим, у випадку коли $p_i = 0$ розповсюдження одиниці до наступних розрядів неможливе (рис. 4).

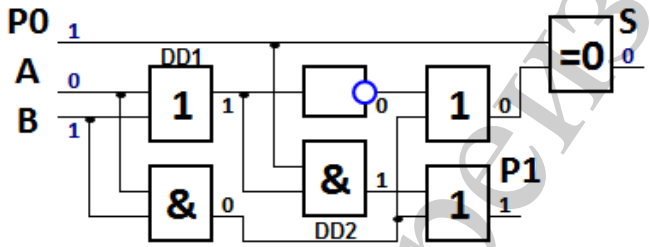

Рис. 4. Демонстрація функції умови перенесення одиниці до старшого розряду

З огляду рис. 4 бачимо, що у випадку коли елемент DD1, який реалізує функцію умови перенесення $p = A \vee B$, на виході отримає значення логічної одиниці ($p = 1$), стане можливим перенесення одиниці P0 до елемента DD2 (вихід DD2=вихід DD1(1)∧P0(1)=1). У випадку, коли на виході елемента DD1 буде логічний нуль, перенесення одиниці P0 до елемента DD2 буде неможливим (вихід DD2=вихід DD1(0)∧P0(1)=0).

Умову перенесення одиниці до старшого розряду можна продемонструвати операцією додавання двійкових чисел у стовпчик (табл. 3).

Таблиця 3

Додавання двійкових чисел у стовпчик

	Можливі варіанти додавання			
Одиниця з молодшого розряду (P_0)	1	1	1	1
Число A	0	0	1	1
Число B	0	1	0	1
Сума	1	10	10	11

З варіантів додавання однорозрядних двійкових чисел бачимо – якщо $A \vee B=1$, одиниця з молодшого розряду P_0 переноситься до старшого (другого) розряду суми ($P_1=1$). Якщо $A \vee B=0$, сума залишається однорозрядною, одиниця з мододшого розряду P_0 до старшого (другого) розряду суми не переноситься ($P_1=0$). Аналогічна умова перенесення одиниці до старшого розряду зберігається і при додаванні багаторозрядних двійкових чисел.

Паралельні префіксні суматори були застосовані в якості найбільш ефективних схем в операціях додавання бінарних кодів цифрових систем. Їх регулярна структура і висока продуктивність зробила їх особливо привабливими для створення НВІС (надвеликих інтегральних схем). Зазначені суматори забезпечують теоретичну базу, для компромісів з точки зору затримки, площі та потужності, з метою подати широкий спектр послуг у процесі проектування.

Паралельний префіксний суматор (PPA) використовує логічну структуру, представлену на рис. 3, яка передбачає обчислення у три етапи:

– попередня обробка (стадія попередньої обробки або ініціалізації):

$$g_i = a_i b_i,$$
$$p_i = a_i \oplus b_i;$$

– розрахунок префіксу (проведення мережі генерації сигналів):

$$G_{[i:k]} = G_{[i:j]} + P_{[i:j]} G_{[j-1:k]},$$
$$P_{[i:k]} = P_{[i:j]} P_{[j-1:k]};$$

– пост-обробка (етап після обробки префікса або підсумовування):

$$C_{i+1} = G_{[i:0]} + P_{[i:0]} \cdot C_0,$$
$$S_i = p_i \oplus C_i.$$

Префіксні архітектури для розрахунку сигналу перенесення відомі, наприклад, такі:

– 1966: Ling adder;
– 1973: Kogge-Stone adder;
– 1980: Ladner-Fisher adder;
– 1982: Brent-Kung adder;
– 1987: Han Carlson adder;
– 1999: S. Knowles. (3)

Серед відомих префіксних структур до основних відноситься паралельний префіксний суматор зі структурою перенесення префікса Лінга та Когге-Стоуна, що є прикінцевим випадком великого переліку підсумовуючих схем, кожна з яких унікальна своєю властивістю мінімальної логічної ємності.

Суматор Лінга (рис. 5) [22–24] має найменшу затримку порівняно з іншими методами перенесення префікса, однак вимагає відносно більшої площі чіпа та енергоспоживання.

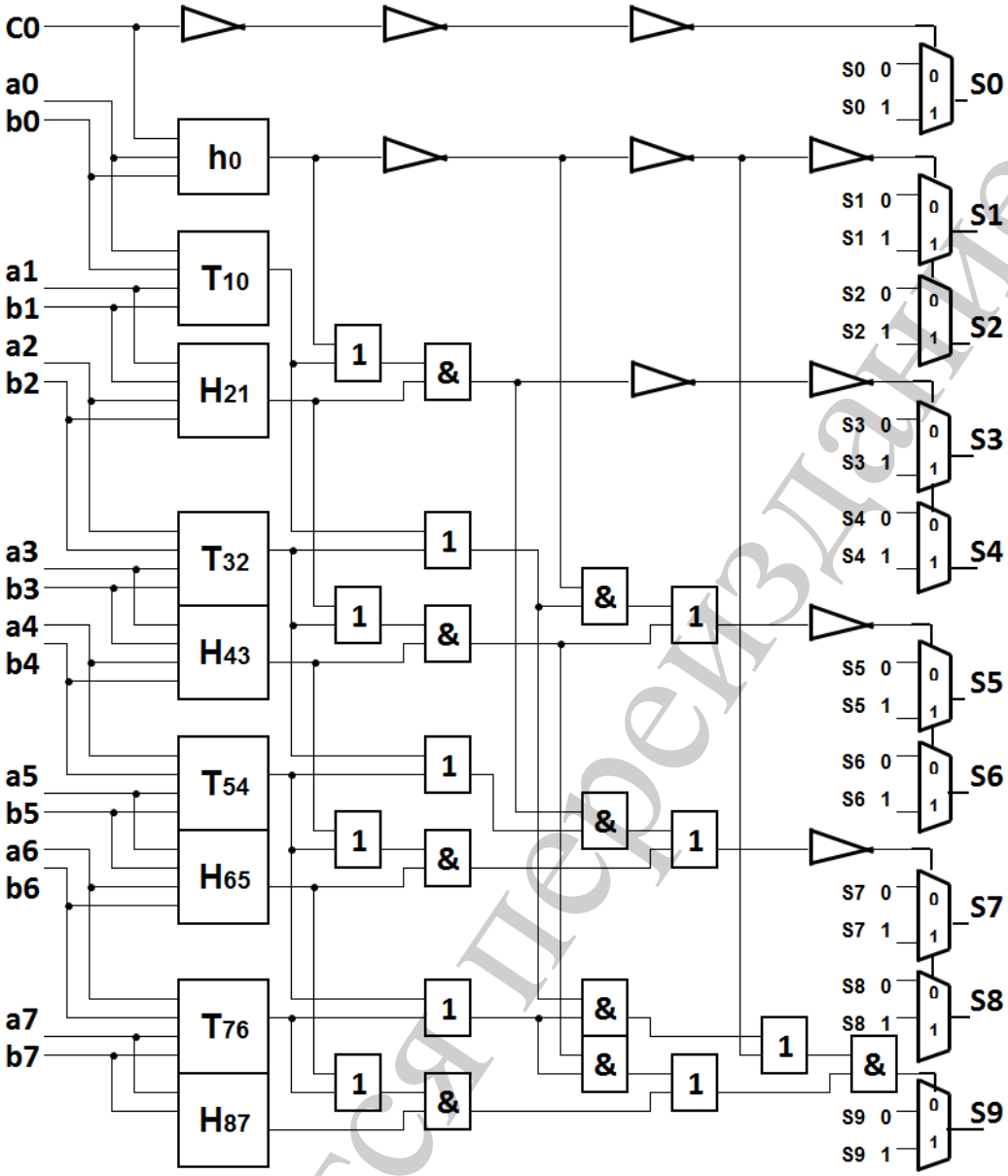

Рис. 5. 8-bit Ling Adder [22–24]

До недоліків префіксної моделі обчислення сигналів суми і перенесення можна віднести:

− процес паралельного обчислення префікса архітектурами (3) передбачає початок обчислення з першого розряду схеми, що приводить, у підсумку, до надлишкового нагромадження і ускладнення апаратної частини пристрою;

− принцип трьохетапного вироблення сигналу суми і перенесення (рис. 3), що задає певну складність такого обчислення, зокрема ускладнює дидактику методу;

− паралельна структура «один до багатьох» префіксного суматора у загальному випадку має менше число зв’язків та може займати декілька рангів схеми. Це, принаймні з технологічної сторони не є ефективним показником, порівняно з ациклічною моделлю обчислення. А від так, можливе протиріччя між вимогами швидкодії обчислення сигналів суми і перенесення та

енергоспоживанням, а також площею пристрою, зокрема у системі дизайну НВІС.

5.2. Ациклічна модель суматора бінарних кодів

Принцип обчислення для моделі ациклічного суматора визначається ациклічним графом, коли одночасно додаються сусідні пари доданків, а потім їх суми (табл. 4). Це є алгоритмом здвоювання (або алгоритмом логарифмічного підсумовування), який у підсумку дає одноетапний спосіб вироблення сигналів суми і перенесення.

Таблиця 4

Алгоритм здвоювання ($n=2^3=8$)

Кроки	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
1	x_1+x_2		x_3+x_4		x_5+x_6		x_7+x_8	
2	$x_1+x_2+x_3+x_4$				$x_5+x_6+x_7+x_8$			
3	$x_1+x_2+x_3+x_4+x_5+x_6+x_7+x_8$							

Проведення порозрядного додавання бінарних кодів можливе за допомогою алгоритму здвоювання, аналогічно процесу багатоопераднього підсумовування. Якщо

$$n=2^k,$$

де n – число доданків, то алгоритм здвоєння складається з k кроків: на першому кроці виконується $n/2$ додавань, на другому – $n/4$, …, на останньому – одне додавання. Кількість кроків k визначається за формулою:

$$k = \log_2 n. \qquad (4)$$

Такий варіант багатоопераднього додавання реалізується за допомогою ациклічного графа або каскадною схемою [21].

Використовуючи процедуру багатоопераднього додавання за допомогою каскадної схеми легко бачити, що для процесу паралельного додавання бінарних кодів парами даних тут будуть біти однойменних розрядів, для кожної з яких обчислюється сигнал суми і перенесення. Далі, аналогічно процедурі багатоопераднього додавання, всі отримані суми однойменних розрядних пар бінарних кодів, зі своєю специфікою, також розбиваються на пари і знову виконується додавання значень пар і т. д.

У підсумку значення старшого розряду суми бінарних кодів можна співставити зі значенням загальної суми при багатоопераднього додаванні. Крім суми старшого розряду у процесі паралельного додавання бінарних кодів автоматично виникають проміжні результати у вигляді значень сум попередніх розрядів бінарних кодів.

Обчислювальна схема паралельного додавання 4-розрядних бінарних кодів може бути визначена орієнтованим ациклічним графом (рис. 1), який являє собою бінарне дерево, де, зокрема, прийняті такі параметри:

k – кількість кроків у часі;

ω – загальна кількість операцій алгоритму;

τ – час виконання одного кроку;

$T=\tau \cdot k$ – час виконання алгоритму;

L – кількість типів операцій та ін.

Обчислювальна схема на рис. 1 представляє також модель 4-розрядного ациклічного паралельного суматора бінарних кодів з паралельним способом перенесення.

Модель обчислювальної схеми ациклічного суматора на рис. 1 використовує дві логічні операції – AND і XOR, число обчислювальних кроків у ній дорівнює розрядності бінарних кодів. Наприклад, для паралельного додавання 4-розрядних бінарних кодів необхідно чотири кроки (рис. 1).

Модель 4-розрядного ациклічного суматора бінарних кодів з логічними елементами OR в останньому розряді представлена на рис. 2.

Застосування ациклічної моделі розраховано на:

− процес послідовного (для молодших розрядів схеми пристрою) і паралельного обчислення сигналів суми і перенесення, що, у підсумку, приводить до зменшення складності апаратної частини пристрою та не збільшує глибину схеми;

− встановлення оптимального числа обчислювальних кроків.

У [21] показано, що число обчислювальних кроків визначає мінімально достатнє число перенесень у схемі ациклічного суматора.

У випадку, коли синтезований суматор отримав більше число перенесень порівняно з числом обчислювальних кроків відповідного орієнтованого ациклічного графа, то такий суматор буде неоптимальним стосовно числа обчислювальних операцій.

Логічні рівняння оптимізованого 4-розрядного суматора з числом перенесення – чотири є, наприклад, такі:

$$S_0 = a_0 \oplus b_0;$$
$$S_1 = (a_1 \oplus b_1) \oplus (a_0 \wedge b_0);$$
$$S_2 = (a_2 \oplus b_2) \oplus ((a_1 \wedge b_1) \vee ((a_1 \vee b_1) \wedge (a_0 \wedge b_0)));$$
$$S_3 = (a_3 \vee b_3) \vee (a_2 \wedge b_2) \vee ((a_2 \vee b_2) \wedge (a_1 \wedge b_1)) \vee$$
$$\vee((a_2 \vee b_2) \wedge ((a_1 \vee b_1) \wedge (a_0 \wedge b_0))).$$

Варіант схеми 4-розрядного ациклічного суматора, яку визначає обчислювальна модель на рис. 2, представлена на рис. 6.

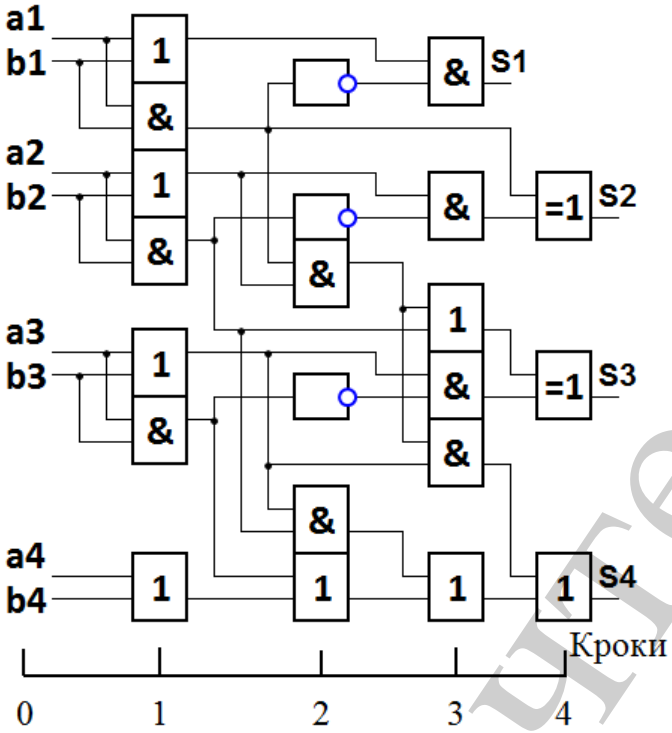

Рис. 6. 4-bit ациклічний суматор бінарних кодів

Модель обчислювальної схеми паралельного 8-розрядного ациклічного суматора бінарних кодів буде подібна обчислювальним схемам, представлених на рис. 1, 2, з тією різницею, що тут число обчислювальних кроків буде дорівнювати восьми. Перші чотири логічні рівняння 8-розрядного ациклічного суматора можуть бути, наприклад, такі:

$$S_0 = a_0 \oplus b_0;$$

$$S_1 = a_0 b_0 \overline{a_1 b_1} + \overline{b_0} a_1 \overline{b_1} + \overline{a_0} a_1 \overline{b_1} + \overline{b_0} a_1 b_1 + \overline{a_0} a_1 b_1 + a_0 b_0 a_1 b_1;$$

$$\begin{aligned} S_2 = {} & a_0 b_0 a_1 \overline{a_2 b_2} + a_0 b_0 \overline{b_1} a_2 \overline{b_2} + a_1 b_1 \overline{a_2 b_2} + \overline{a_1 b_1} a_2 \overline{b_2} + \\ & + \overline{b_0 b_1} a_2 \overline{b_2} + \overline{a_0 b_1} a_2 \overline{b_2} + \overline{b_0 a_1} a_2 \overline{b_2} + \overline{a_0 a_1} a_2 \overline{b_2} + \overline{a_1 b_1} a_2 b_2 + \\ & + \overline{b_0 b_1} a_2 b_2 + \overline{a_0 b_1} a_2 b_2 + \overline{b_0 a_1} a_2 b_2 + \overline{a_0 a_1} a_2 b_2 + \\ & + a_0 b_0 a_1 a_2 b_2 + a_0 b_0 b_1 a_2 b_2 + a_1 b_1 a_2 b_2; \end{aligned}$$

$$S_3 = a_0b_0a_1a_2\overline{a_3b_3} + a_0b_0b_1a_2\overline{a_3b_3} + a_1b_1a_2\overline{a_3b_3} +$$
$$+a_0b_0a_1b_2\overline{a_3b_3} + a_0b_0b_1b_2\overline{a_3b_3} + a_1b_1b_2\overline{a_3b_3} + a_2b_2\overline{a_3b_3} +$$
$$+\overline{a_2b_2}a_3\overline{b_3} + \overline{a_1b_1b_2}a_3\overline{b_3} + \overline{b_0b_1b_2}a_3\overline{b_3} + \overline{a_0b_1b_2}a_3\overline{b_3} +$$
$$+\overline{b_0a_1b_2}a_3\overline{b_3} + \overline{a_0a_1b_2}a_3\overline{b_3} + \overline{a_1b_1a_2}a_3\overline{b_3} + \overline{b_0b_1a_2}a_3\overline{b_3} +$$
$$+\overline{a_0b_1a_2}a_3\overline{b_3} + \overline{b_0a_1a_2}a_3\overline{b_3} + \overline{a_0a_1a_2}a_3\overline{b_3} +$$
$$+\overline{a_2b_2}a_3b_3 + \overline{a_1b_1b_2}a_3b_3 + \overline{b_0b_1a_2}a_3b_3 + \overline{a_0b_1b_2}a_3b_3 +$$
$$+\overline{b_0a_1b_2}a_3b_3 + \overline{a_0a_1b_2}a_3b_3 + \overline{a_1b_1a_2}a_3b_3 + \overline{b_0b_1a_2}a_3b_3 +$$
$$+\overline{a_0b_1a_2}a_3b_3 + \overline{b_0a_1a_2}a_3b_3 + \overline{a_0a_1a_2}a_3b_3 + a_0b_0a_1a_2a_3b_3 +$$
$$+a_0b_0b_1a_2a_3b_3 + a_1b_1a_2a_3b_3 + a_0b_0a_1b_2a_3b_3 + a_0b_0b_1b_2a_3b_3 +$$
$$+a_1b_1b_2a_3b_3 + a_2b_2a_3b_3.$$

Схеми 8-bit ациклічних суматорів, представлені на рис. 8, 10, 12. Зі збільшенням розрядності ациклічного суматора (16-, 32-, 64-bit …) число обчислювальних кроків буде визначатись за логарифмічним законом (рис. 7).

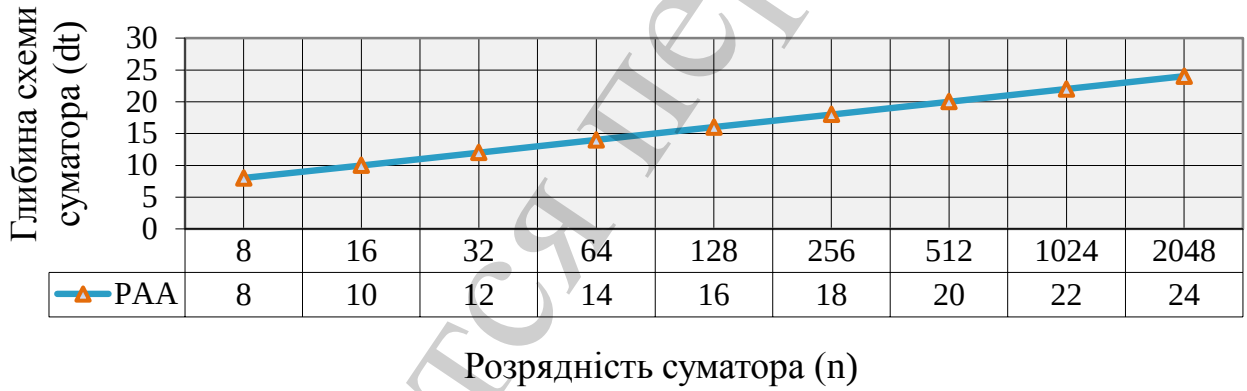

Рис. 7. Динаміка збільшення глибини схеми ациклічного суматора (PAA)

6. Результати дослідження

6.1. 8-bit ациклічний суматор з глибиною схеми 8 елементів

Для забезпечення однакових умов порівняння будемо представляти схеми префіксних (PPA) та ациклічних (PAA) суматорів з логічними елементами OR в останньому розряді.

На рис. 8 представлений ациклічний 8-bit PAA з логічними елементами OR в останньому розряді та глибиною схеми 8 типових 2-входових елементів. Складність схеми на рис. 8 становить 77 дискретних елементи.

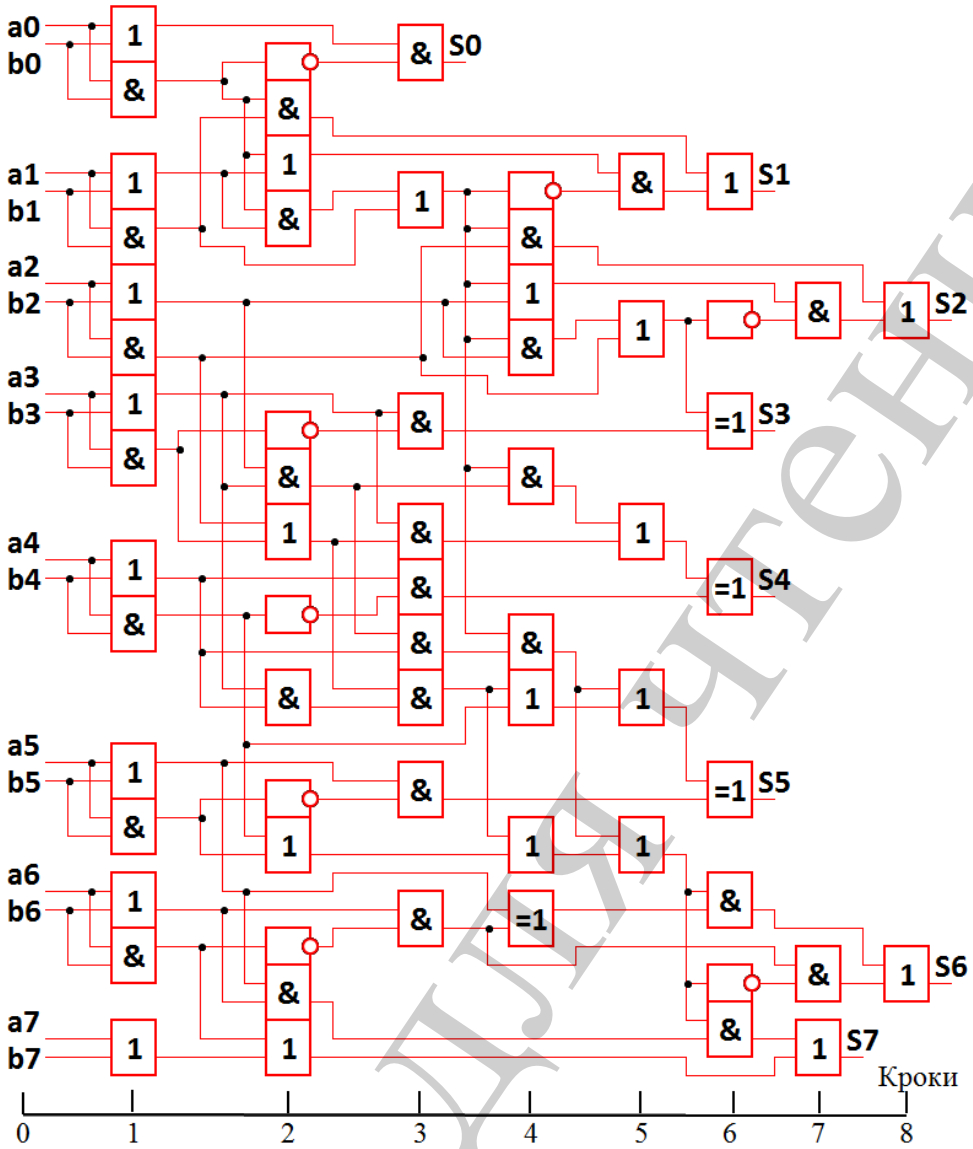

Рис. 8. Ациклічний 8-bit PAA з логічними елементами OR в останньому розряді та глибиною схеми 8 типових 2-входових елементів

Префіксний 8-bit Ling Adder [22–24] з логічними елементами OR в останньому розряді представлений на рис. 9. Ланцюг, що визначає глибину схеми суматора на рис. 9 виділений жирною лінією та супроводжується нумерацією логічних елементів уздовж зазначеного ланцюга. Таким чином глибина схеми 8-bit Ling Adder [22–24] PPA (рис. 9) складає 8 типових логічних елементів, складність схеми становить 109 елементів. У відповідності зі структурою префіксної моделі (рис. 3), третій етап обчислення сигналу суми в суматорі на рис. 9 реалізується мультиплексором у кожному розряді схеми.

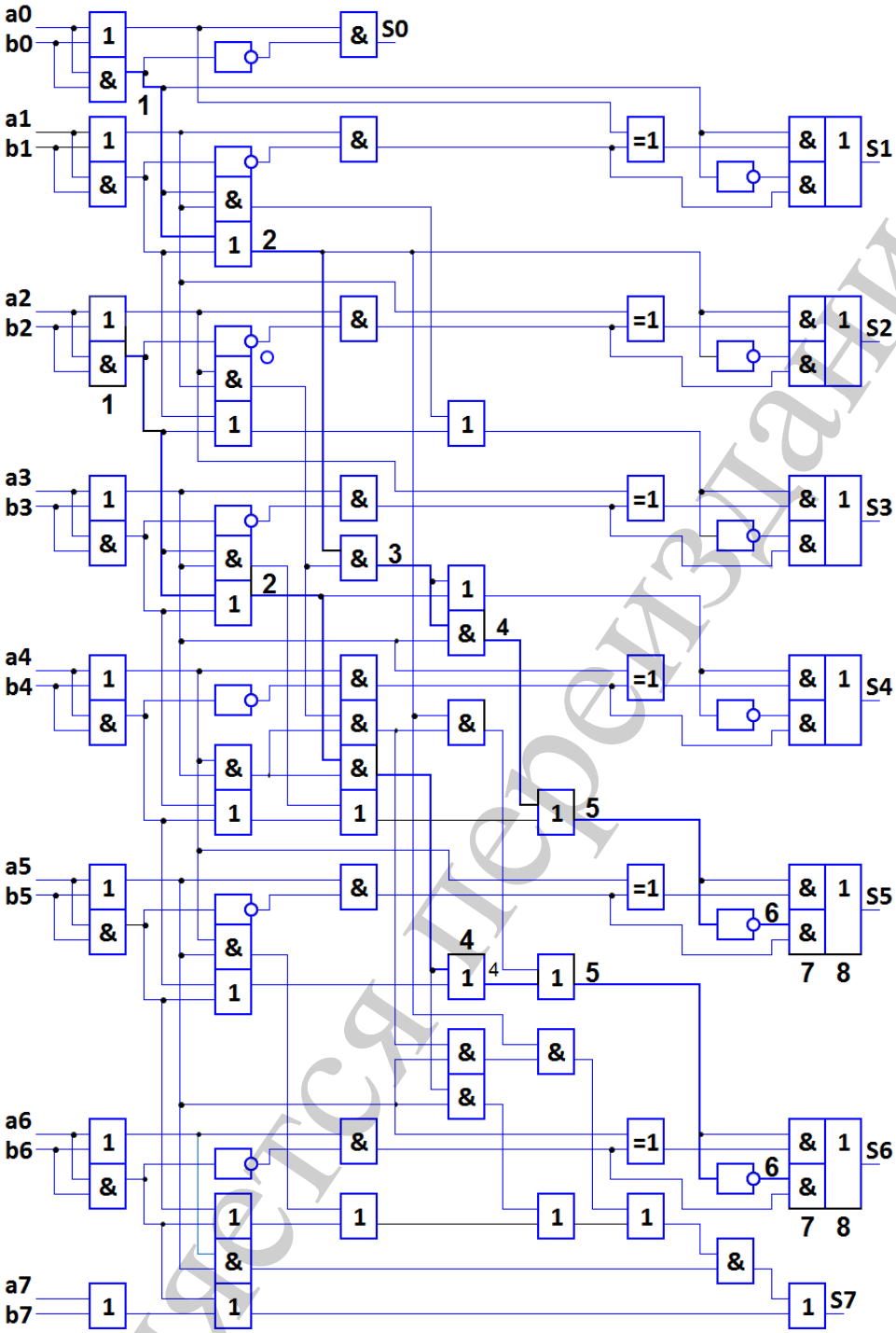

Рис. 9. Префіксний 8-bit Ling Adder PPA [22–24] з логічними елементами OR в останньому розряді

Обчислювальний процес суматора 8-bit Ling Adder PPA (рис. 9) використовує такі логічні операції: XOR – 13, AND – 27, OR – 24, Inventor – 6. Суматор 8-bit PAA (рис. 8) використовує: XOR – 9, AND – 19, OR – 19, Inventor – 3. Враховуючи те, що логіка елемента XOR використовує чотири логічних елементи, включаючи Inventor можна оцінити показник якості *S* (наприклад, стосовно енергозбереження) роботи суматора 8-bit PAA (рис. 8):

$$S = \frac{T_1}{T_2} = \frac{109}{77} = 1,4156 = 41,56\ \%,$$

де T_1, T_2 – число дискретних логічних елементів 8-bit Ling Adder PPA та 8-bit PAA відповідно.

6.2. 8-bit ациклічний суматор з глибиною схеми 9 елементів

На рис. 10 представлений ациклічний 8-bit PAA з логічними елементами OR в останньому розряді та глибиною схеми 9 типових 2-входових елементів. Складність схеми на рис. 10 становить 72 дискретних елементи.

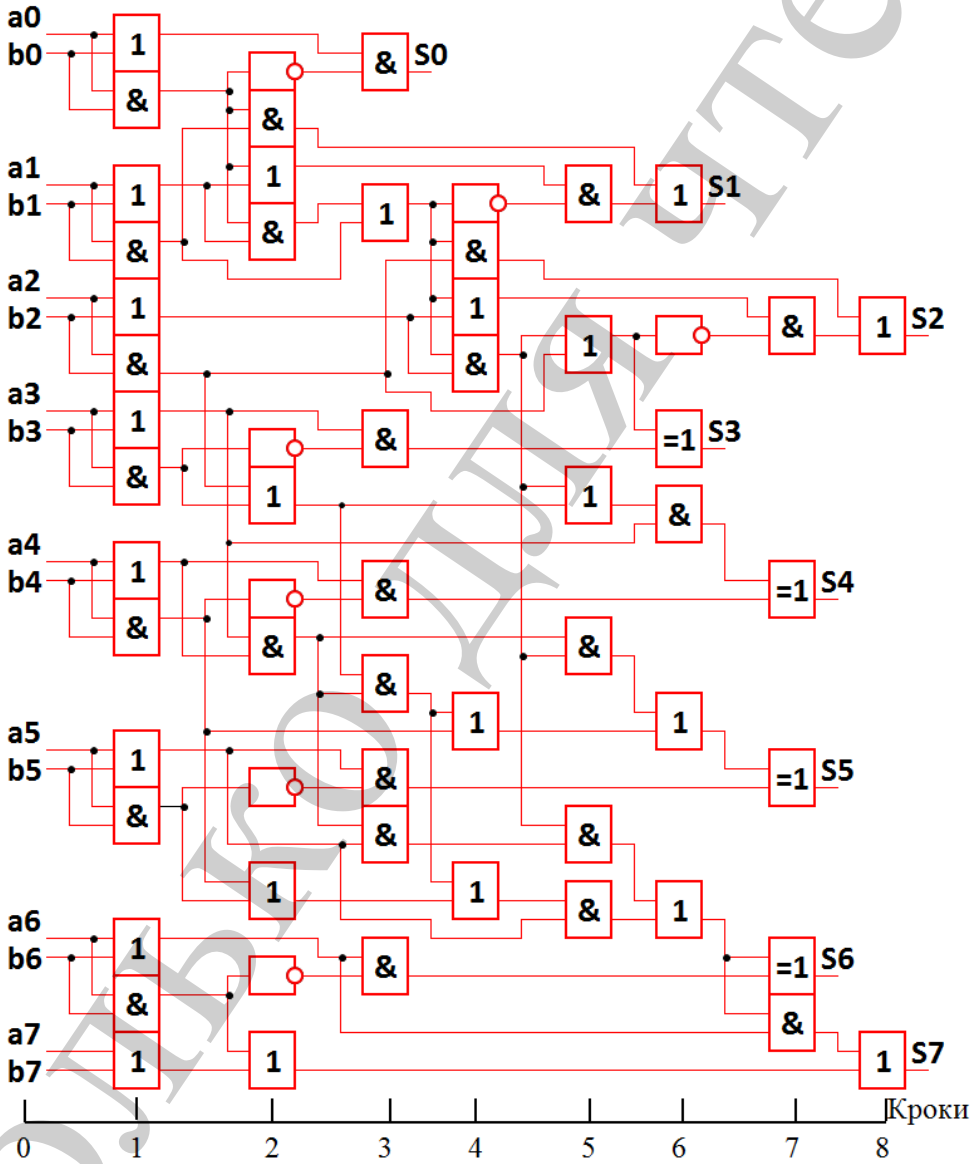

Рис. 10. Ациклічний 8-bit PAA з логічними елементами OR в останньому розряді та глибиною схеми 9 типових 2-входових елементів

Префіксний 8-bit Kogge-Stone PPA [25] з логічними елементами OR в останньому розряді представлений на рис. 11. Ланцюг, що визначає глибину схеми суматора на рис. 11 виділений жирною лінією та супроводжується

нумерацією логічних елементів уздовж зазначеного ланцюга. Таким чином глибина схеми 8-bit Kogge-Stone PPA (рис. 11) складає 9 типових логічних елементів, складність схеми становить 90 елементів.

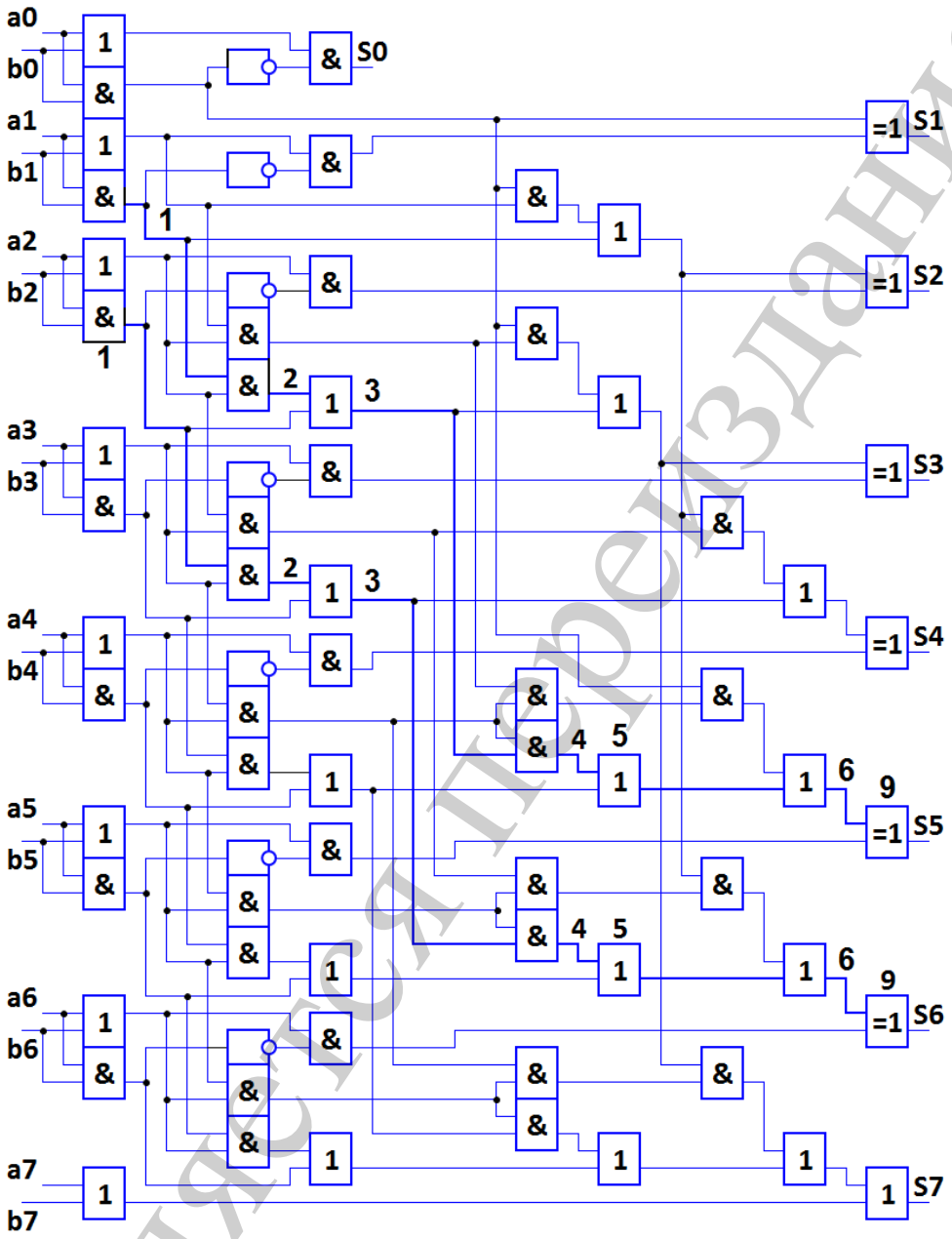

Рис. 11. Префіксний 8-bit Kogge-Stone PPA з логічними елементами OR в останньому розряді [25]

Обчислювальний процес суматора 8-bit Kogge-Stone PPA (рис. 11) використовує такі логічні операції :XOR – 13, AND – 22, OR – 26. Суматор 8-bit PAA (рис. 10) використовує: XOR – 9, AND – 16, OR – 18, Inventor – 2. Показник якості *S* (наприклад, стосовно енергозбереження) роботи суматора 8-bit PAA (рис. 10) є таким:

$$S = \frac{T_1}{T_2} = \frac{90}{72} = 1,25 = 25\ \%,$$

де T_1, T_2 – число дискретних логічних елементів 8-bit Kogge-Stone PPA та 8-bit PAA відповідно.

6.3. 8-bit ациклічний суматор з глибиною схеми 10 елементів

На рис. 12 представлений ациклічний 8-bit PAA з логічними елементами OR в останньому розряді та глибиною схеми 10 типових 2-входових елементів. Складність схеми на рис. 12 становить 66 дискретних елементи.

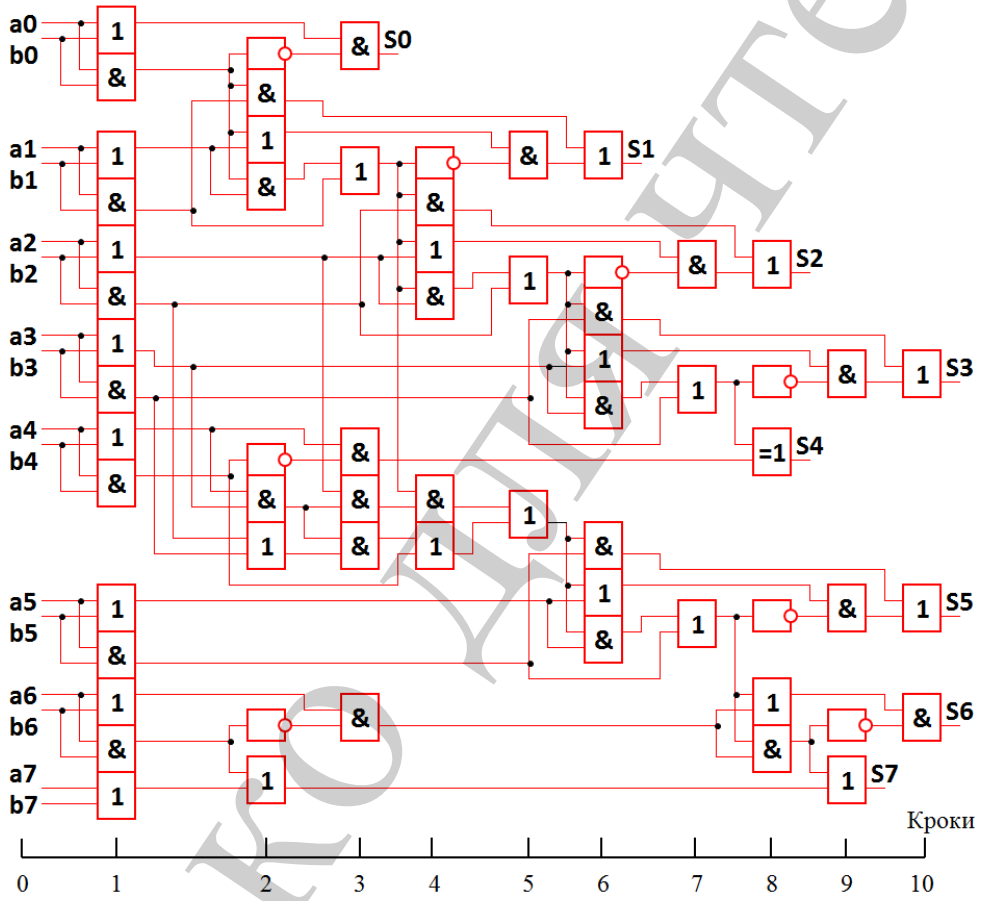

Рис. 12. Ациклічний 8-bit PAA з логічними елементами OR в останньому розряді та глибиною схеми 10 типових 2-входових елементів

Префіксний 8-bit Brent-Kung PPA [25] з логічними елементами OR в останньому розряді представлений на рис. 13.

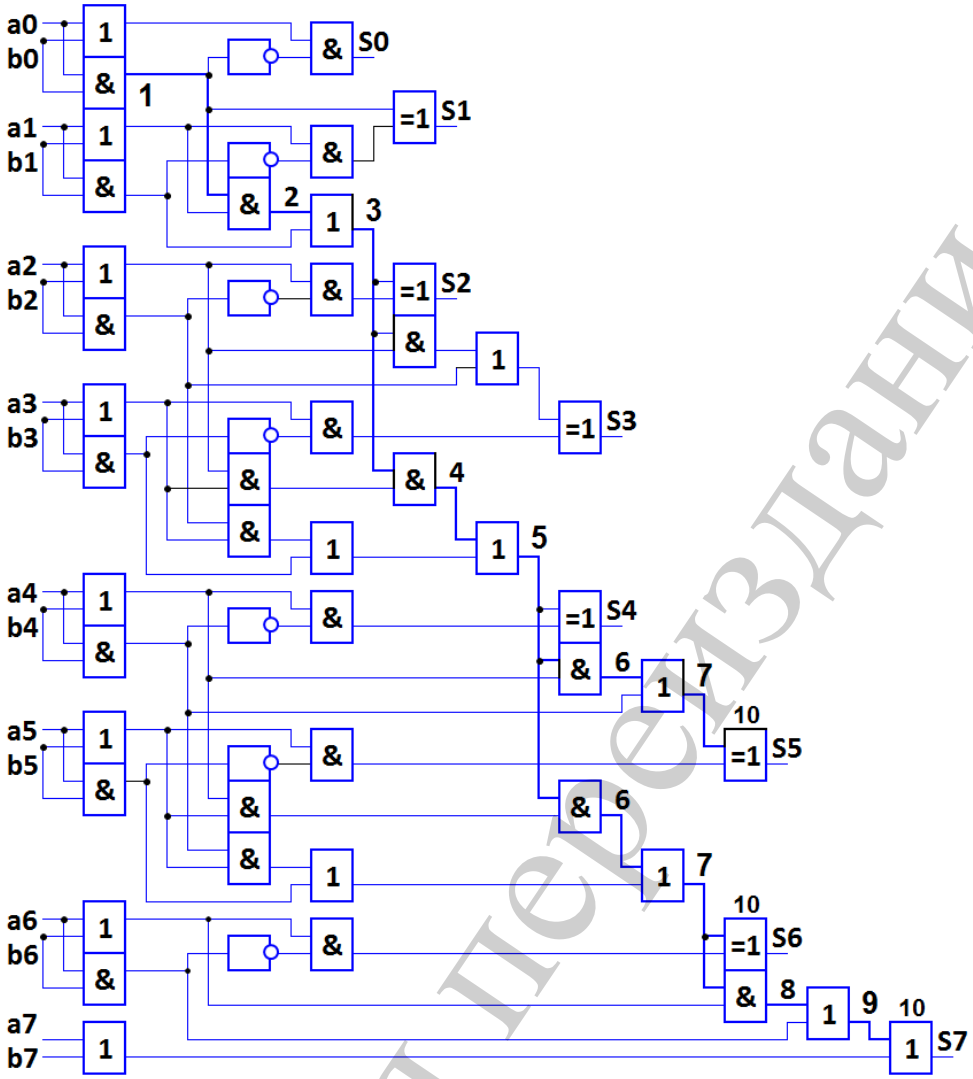

Рис. 13. Префіксний 8-bit Brent-Kung PPA з логічними елементами OR в останньому розряді [25]

Ланцюг, що визначає глибину схеми суматора на рис. 13 виділений жирною лінією та супроводжується нумерацією логічних елементів уздовж зазначеного ланцюга. Таким чином глибина схеми 8-bit Brent-Kung PPA (рис. 13) складає 10 типових логічних елементів, складність схеми становить 72 дискретних елементи. Зазначимо, що елемент XOR має глибину схеми три дискретних елементи та складається з чотирьох дискретних логічних елементів, включаючи Inventor.

Обчислювальний процес суматора 8-bit Brent-Kung PPA (рис. 13) використовує такі логічні операції: XOR – 13, AND – 10, OR – 10. Суматор 8-bit PAA (рис. 12) використовує: XOR – 5, AND – 20, OR – 22, Inventor – 4. Показник якості S (наприклад, стосовно енергозбереження) роботи суматора 8-bit PAA (рис. 12) є таким:

$$S = \frac{T_1}{T_2} = \frac{72}{66} = 1{,}0909 = 9{,}09\ \%,$$

де T_1, T_2 – число дискретних логічних елементів 8-bit Brent-Kung PPA та 8-bit PAA відповідно.

6.4. Порівнянний аналіз ациклічної та префіксної моделей обчислення сигналів суми і перенесення

Параметри синтезованих схем ациклічних та префіксних суматорів зведені до порівняльної табл. 5.

Таблиця 5

Порівняльна таблиця параметрів префіксних та ациклічних суматорів

Паралельний суматор бінарних кодів з паралельним перенесенням		Глибина схеми	Складність схеми	Розряд-ність суматора
Ациклічний суматор	рис. 8	8	77	8-bit
Префіксний суматор	Ling Adder (рис.9)	8	109	8-bit
Ациклічний суматор	рис. 10	9	72	8-bit
Префіксний суматор	Kogge-Stone (рис.11)	9	90	8-bit
Ациклічний суматор	рис. 12	10	66	8-bit
Префіксний суматор	Brent-Kung (рис. 13)	10	72	8-bit

З огляду табл. 5 бачимо, що при обраному значенні глибини схеми, складність схем ациклічних суматорів є меншою.

Показники якості ациклічних суматорів, наприклад, за енергоспоживанням представлені у табл. 6.

Таблиця 6

Порівняльна таблиця показників якості за енергоспоживанням префіксних та ациклічних суматорів

Паралельний суматор бінарних кодів з паралельним перенесенням		Показник якості ациклічного суматора
Ациклічний суматор	рис. 8	41,56 %
Префіксний суматор	Ling Adder (рис.9)	
Ациклічний суматор	рис. 10	25 %
Префіксний суматор	Kogge-Stone (рис.11)	
Ациклічний суматор	рис. 12	9,09 %
Префіксний суматор	Brent-Kung (рис. 13)	

На рис. 14 представлена динаміка збільшення глибини схеми для трьох ациклічних суматорів (PAA) (рис. 8, 10, 12) зі збільшенням розрядності схеми.

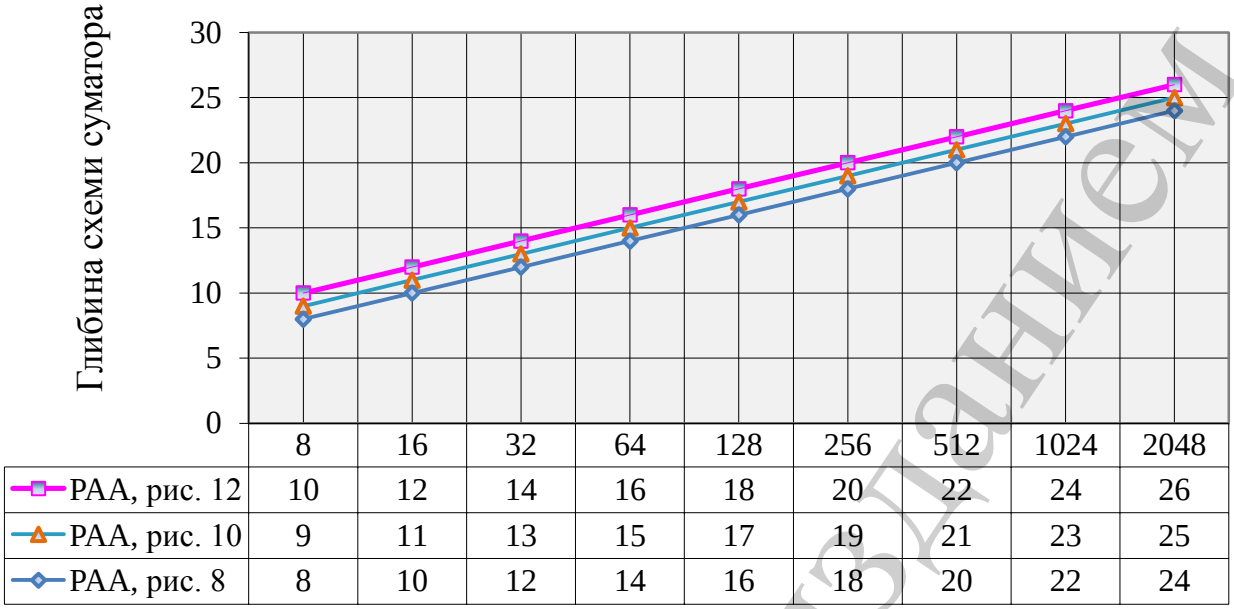

Рис. 14. Динаміка збільшення глибини схеми ациклічних суматорів (РАА)

У табл. 7 представлено порівняння префіксної [1–4] та ациклічної моделей обчислення сигналів суми і перенесення у схемі суматора.

Таблиця7

Порівняльна таблиця двох моделей обчислення сигналів суми і перенесення

Префіксна модель	Ациклічна модель
Спосіб обчислення префікса	
Префікса модель передбачає процес обчислення префікса, починаючи з першого розряду схеми, що приводить, у підсумку, до надлишкового нагромадження і ускладнення апаратної частини пристрою	Застосування ациклічної моделі розраховано на: − логічну структуру схеми суматора з послідовно-паралельним способом обчислення префікса, що, у підсумку, дає зменшення складності апаратної частини пристрою та не збільшує глибину схеми; − встановлення оптимального числа обчислювальних кроків
Кількість етапів обчислення	
Префіксна модель використовує три етапи вироблення сигналів суми і перенесення (рис. 2)	Ациклічна модель використовує один етап вироблення сигналів суми і перенесення (рис. 4, 5)
Показник паралельності структури суматора	
Паралельна структура «один до багатьох» префіксного суматора (рис. 9, 11, 13) у загальному випадку має менше число зв'язків, порівняно з ациклічним суматором	Паралельна структура «один до багатьох» ациклічного суматора (рис. 8, 10, 12) у загальному випадку має більше число зв'язків, порівняно з префіксним суматором, що засвідчує більшу ступінь паралельності схеми ациклічного суматора

З огляду табл. 7 випливає, що ациклічна модель обчислення сигналів суми і перенесення для схем суматорів бінарних кодів заперечує префіксну модель обчислення.

7. SWOT-аналіз результатів досліджень

Strengths. До сильної сторони ациклічної моделі обчислення сигналів суми і перенесення можна віднести дидактичні спрощення та апаратну компактність методу, що дозволяє замінити трьохетапну префіксну модель на одноетапну ациклічну модель обчислення сигналів суми і перенесення. Це дасть розширення апарату синтезу арифметичних пристроїв для їхнього застосування у цифрових технологіях.

Зв'язок між числом обчислювальних кроків орієнтованого ациклічного графа і числом перенесень одиниці до старшого розряду спричиняє процес співставлення структури суматора з відповідним орієнтованим ациклічним графом. Метою зазначеного співставлення є встановлення мінімально достатнього числа перенесень для операції додавання двійкових чисел у схемі паралельного суматора з паралельним способом перенесення. У випадку, коли синтезований суматор отримав більше число перенесень порівняно з числом обчислювальних кроків відповідного орієнтованого ациклічного графа, то такий суматор буде неоптимальним стосовно числа обчислювальних операцій.

Ациклічна модель спроможна підтримувати агреговані структури обчислення сигналів суми і перенесення, шляхом об'єднання з відповідним апаратом інших методів обчислення, зокрема з логікою перенесення Лінга.

Це вигідніше у порівнянні з аналогами за такими чинниками:

− меншою вартістю розробки та впровадження, оскільки ациклічна модель визначає порівняно простішу структуру суматора;

− наявністю критерію оптимізації – число обчислювальних кроків ациклічного графа вказує на мінімально достатнє число перенесень одиниці до старшого розряду.

Weaknesses. Слабка сторона ациклічної моделі обчислення сигналів суми і перенесення пов'язана зі зростанням трудомісткості синтезу обчислювальної структури та недостатнім вивченням такого синтезу зі збільшенням розрядності схеми пристрою.

Негативні внутрішні фактори притаманні ациклічній моделі полягають у збільшенні часу отримання оптимальної структури обчислення при зростанні розрядності схеми суматора.

Opportunities. Перспективою подальших досліджень ациклічної моделі може бути вироблення протоколу оптимального чергування логіки перенесення Лінга та логіки перенесення ациклічної моделі з метою зменшення складності схеми суматора.

Додаткові можливості, що можуть принести впровадження ациклічної моделі, полягають у вивченні варіантів застосування функції умови перенесення одиниці до старшого розряду (1). Це дасть можливість отримувати оптимальну складність обчислювальної структури арифметичного пристрою.

Threats. Протокол обчислення сигналів суми та перенесення ациклічної моделі є незалежним від протоколів інших методів обчислення, тому загроза негативної дії на об'єкт дослідження зовнішніх чинників відсутня.

До певної міри аналогом ациклічної моделі синтезу схеми суматора є префіксна модель. На даний момент префіксна модель краща тим, що за її допомогою вже створені та впроваджені арифметичні пристрої з префіксною структурою обчислення.

8. Висновки

1. Виявлено, що обчислення сигналу суми і перенесення у схемі паралельного ациклічного суматора здійснюється за алгоритмом логарифмічного додавання. Число обчислювальних кроків ациклічного графа визначає оптимальне число перенесень у схемі паралельного суматора з паралельним способом перенесення.

2. Оцінка динаміки збільшення глибини схеми ациклічного суматора складає $O(n)$ і є лінійною для $n \leq 8$. Зі збільшенням розрядності схеми від $n>8$ оцінка динаміки збільшення глибини схеми ациклічного суматора складає $O(\log n)$ і є логарифмічною.

3. Ефективність ациклічної моделі демонструється прикладами синтезу 8-розрядних паралельних суматорів, запозичених з робіт інших авторів з метою порівняння:

− схема суматора Лінга (рис. 9) [22–24] та схема ациклічного 8-розрядного паралельного суматора з глибиною схеми 8 елементів (рис. 8);

− схема префіксного суматора Kogge-Stone (рис. 11) [25] та схема ациклічного 8-розрядного паралельного суматора з глибиною схеми 9 елементів (рис. 10);

− схема префіксного суматора Brent-Kung (рис. 13) [25] та схема ациклічного 8-розрядного паралельного суматора з глибиною схеми 10 елементів (рис. 12).

З огляду на зазначені приклади паралельних суматорів, ациклічна модель дає підставу для доцільності її застосування у процесах синтезу арифметичних пристроїв обробки цифрових даних, оскільки зазначені схеми спроможні:

− збільшити швидкодію;

− зменшити енергоспоживання та тепловиділення цифрового пристрою, інтегральної схеми.

Література

1. Brent R. P., Kung H. T. A regular layout for parallel adders // IEEE Transactions on Computers. 1982. Vol. 31, No. 3. P. 260–264. doi: http://doi.org/10.1109/tc.1982.1675982

2. Han T., Carlson D. A. Fast area-efficient VLSI adders // IEEE 8th Symposium on Computer Arithmetic (ARITH). 1987. doi: http://doi.org/10.1109/arith.1987.6158699

3. Kogge P. M., Stone H. S. A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations // IEEE Transactions on Computers. 1973. Vol. 22, No. 8. P. 786–793. doi: http://doi.org/10.1109/tc.1973.5009159

4. Ladner R. E., Fischer M. J. Parallel Prefix Computation // Journal of the ACM. 1980. Vol. 27, No. 4. P. 831–838. doi: http://doi.org/10.1145/322217.322232

5. Solomko M., Olshansky P. The Parallel Acyclic Adder // 2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM). Lviv, 2017. P. 125–129.

6. Mathematical Modeling of Timing Attributes of Self-Timed Carry Select Adders / Balasubramanian P. et al. // Recent Advances in Circuits, Systems, Telecommunications and Control. 2013. P. 228–243. URL: http://www.wseas.us/e-library/conferences/2013/Paris/CCTC/CCTC-34.pdf

7. Venkatanaga Kumar G., Pushpalatha C. H. Implementation of Carry Tree Adders and Compare with RCA and CSLA // International Journal of Emerging Engineering Research and Technology. 2016. Vol. 4, No. 1. P. 1–11. URL: http://www.ijeert.org/pdf/v4-i1/1.pdf

8. Gedam K. S., Zode P. P. Parallel prefix han-carlson adder // International Journal of Research in Engineering and Applied Sciences. 2014. Vol. 2, No. 2. P. 81–84. URL: http://mgijournal.com/pdf_new/electronics/swapna%20gedam-1.pdf

9. Krishna Kumari V., Sri Chakrapani Y., Kamaraju M. Design and Characterization of Kogge-Stone, Sparse Kogge-Stone, Spanning tree and Brent-Kung Adders // International Journal of Scientific & Engineering Research. 2013. Vol. 4, No. 10. P. 1502–1506. URL: https://www.ijser.org/researchpaper/Design-and-Characterization-of-Koggestone-Sparse-Koggestone-Spanning-tree-and-Brentkung-Adders.pdf

10. Ramanathan P., Vanathi P. T. Hybrid Prefix Adder Architecture for Minimizing the Power Delay Product // World Academy of Science, Engineering and Technology International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering. 2009. Vol. 3, No. 4. P. 869–873. URL: https://waset.org/publications/5272/hybrid-prefix-adder-architecture-for-minimizing-the-power-delay-product

11. Kaarthik K., Vivek C. Hybrid Han Carlson Adder Architecture for Reducing Power and Delay Middle-East // Journal of Scientific Research. 2016. Vol. 24. P. 308–313. URL: https://www.idosi.org/mejsr/mejsr24(IIECS)16/48.pdf

12. Yagain D., Vijaya K. A., Baliga A. Design of High-Speed Adders for Efficient Digital Design Blocks. ISRN Electronics. 2012. Vol. 2012. P. 1–9. doi: http://doi.org/10.5402/2012/253742

13. Krishna B., Siva Durga Rao P., Prasad N. V. G. High Speed and Low Power Design of Parallel Prefix Adder // International Journal of Electronics & Communication Technology. 2012. Vol. 3, No. 4. P. 472–475. URL: http://www.iject.org/vol34/3/a572

14. Aktan M., Baran D., Oklobdzija V. G. Minimizing Energy by Achieving Optimal Sparseness in Parallel Adders // 2015 IEEE 22nd Symposium on Computer Arithmetic. 2015. P. 10–17. doi: http://doi.org/10.1109/arith.2015.13

15. Anitha R., Bagyaveereswaran V. High performance parallel prefix adders with fast carry chain logic // International Journal of Advanced Research in Engineering and Technology (IJARET). 2012. Vol. 3, No. 2. URL: https://www.slideshare.net/iaemedu/high-performance-parallel-prefix-adders-with-fast-carry-chain-logic

16. Kombinatsiinyi sumator: Patent 115751 UA, MPK G 06 F 7/501 (2006.01) / Vozna N. Ya. et al. Appl. No. a201701347; Filed: 13.02.2017; Published: 11.12.2017; Bul. No. 23. URL: http://uapatents.com/6-115751-kombinacijjnijj-sumator.html

17. Gurkayna F. K. et al. Higher radix Kogge-Stone parallel prefix adder architectures // 2000 IEEE International Symposium on Circuits and Systems. Emerging Technologies for the 21st Century. Proceedings (IEEE Cat No. 00CH36353). Presses Polytech. Univ. Romandes, 2000. doi: http://doi.org/10.1109/iscas.2000.857516

18. Knowles S. A family of adders // Proceedings 14th IEEE Symposium on Computer Arithmetic (Cat. No. 99CB36336). IEEE Comput. Soc., 1999. doi: http://doi.org/10.1109/arith.1999.762825

19. Beaumont-Smith A., Lim C.-C. Parallel prefix adder design // Proceedings 15th IEEE Symposium on Computer Arithmetic. ARITH-15 2001. IEEE Comput. Soc., 2001. doi: http://doi.org/10.1109/arith.2001.930122

20. Zimmermann R. Efficient VLSI implementation of modulo (2/sup n/±1) addition and multiplication // Proceedings 14th IEEE Symposium on Computer Arithmetic (Cat. No. 99CB36336). IEEE Comput. Soc., 1999. doi: http://doi.org/10.1109/arith.1999.762841

21. Zeydel B. R., Baran D., Oklobdzija V. G. Energy-Efficient Design Methodologies: High-Performance VLSI Adders // IEEE Journal of Solid-State Circuits. 2010. Vol. 45, No. 6. P. 1220–1233. doi: http://doi.org/10.1109/jssc.2010.2048730

22. Govindarajulu S., Vijaya Durga Royal T. Design of Energy-Efficient and High-Performance VLSI Adders // International Journal of Engineering Research. 2014. Vol. 3, No. 2, P. 55–59. URL: http://ijer.irponline.in/ijer/publication/v3si2/IJER_2014_NCSC%2013.pdf

23. Pinto R., Shama K. Efficient shift-add multiplier design using parallel prefix adder // International Journal of Control Theory and Applications. 2016. Vol. 9, No. 39. P. 45–53. URL: http://serialsjournals.com/serialjournalmanager/pdf/1500377875.pdf

24. Solomko M., Krulikovskyi B. Study of carry optimization while adding binary numbers in the rademacher number-theoretic basis // Eastern-European Journal of Enterprise Technologies. 2016. Vol. 3, No. 4 (81). P. 56–63. doi: http://doi.org/10.15587/1729-4061.2016.70355